AI TECHNIQUES FOR A SPACE APPLICATION SCHEDULING PROBLEM

N. Thalman, T. Sparn, L. Jaffres, D. Gablehouse, D. Judd, C. Russell
Laboratory for Atmospheric and Space Physics
University of Colorado
Boulder, Colorado 80309

ABSTRACT

Scheduling is a very complex optimization problem which can be categorized as an NP-complete problem. NP-complete problems are quite diverse, as are the algorithms used in searching for an optimal solution. In most cases, the best solutions that can be derived for these combinatorial explosive problems are near-optimal solutions. Due to the complexity of the scheduling problem, artificial intelligence (AI) techniques can aid in solving these types of problems. This paper examines some of the factors which make space application scheduling problems difficult and presents a fairly new AI-based technique called tabu search as applied to a real scheduling application. The specific problem is concerned with scheduling solar and stellar observations for the SOLar-STellar Irradiance Comparison Experiment (SOLSTICE) instrument in a constrained environment which produces minimum impact on other instruments and maximizes target observation times. The SOLSTICE instrument will fly on-board the Upper Atmosphere Research Satellite (UARS) in 1991, and a similar instrument will fly on the Earth Observing System (EOS).

INTRODUCTION

As space applications become more operationally complex, automated scheduling systems are a necessity. Missions such as the Hubble Space Telescope, the Upper Atmosphere Research Satellite, and the Earth Observing System will require some level of automated scheduling in order for activities to proceed and resources to be used in a safe and efficient manner. Increasingly, prototyped and operational scheduling systems using artificial intelligence techniques to aid in search and optimization are proving to be successful. The difficulty is to apply these techniques in an efficient manner when solving scheduling problems.

In space applications which have scientific goals, a scheduling system provides a bridge between the science environment and the resource environment. In the operational context it is important to maximize activities with respect to resources. At the same time, scientific objectives must be fulfilled. Coupling these contexts will allow for the maximizing of activities and scientific return with respect to available resources and the resultant schedule will satisfy science objectives.

Research to find good solutions for the scheduling problem includes studies of classical industrial problems involving tasks with predefined precedences and deadlines (such as the Flow-shop, Job-shop or Travelling Salesman problems). Such problems can be solved by general conventional techniques like Branch-and-Bound, PERT or Critical Path Method (CPM) [Wiest and Levy, 1969] [Kaufmann and Desbazeille, 1969]. However, using conventional techniques to solve resource-based scheduling problems for space applications is not easily accomplished.

Scheduling in the space application environment is similar to the factory scheduling problem because it is concerned with the allocation of a limited amount of resources to specific operations over time. Large numbers of activities, resources, and interacting relationships complicate these problems. Procedures that guarantee an optimum solution to scheduling problems are either NP-complete or NP-hard [Ibaraki and Katoh, 1988], most frequently NP-hard in realistic cases. Combinatorially complex problems such as resource-based constraint scheduling are NP-hard. In theory, NP-hard problems have no polynomial algorithm for finding optimal solutions or the number of possible solutions increases so fast that there are no practical results. In most cases, the best solutions for these problems are near optimal. The difficulty of the problem remains mainly in the large number of constraints which have to be satisfied at the same time.

In recent years, artificial intelligence techniques such as neural networks, simulated annealing and genetic algorithms have been prototyped and demonstrated to aid in the solution of resource-based constraint scheduling problems. Another technique gaining recognition is tabu search. Tabu search is a proven search technique for scheduling problems and offers advantages over other techniques with regard to ease of implementation and the flexibility to handle additional considerations not encompassed by the original problem. Some of the applications where tabu search has had great success in terms of performance and adaptability include employee scheduling, space planning and architectural design, travelling salesman problems, and job-shop scheduling problems [Glover, 1990].

The first section of this paper describes factors which make the resource-based constraint scheduling problem difficult. The second section describes the UARS SOLSTICE scheduling problem, followed by a third section which discusses AI-based techniques examined for use on this flight project. The fourth section explains the AI-based tabu search methodology which has been selected for the UARS SOLSTICE scheduling application. And the fifth section discusses the current knowledge-based solution of the UARS SOLSTICE scheduling application and compares it to the tabu search implementation.

FACTORS IN RESOURCE-BASED CONSTRAINT SCHEDULING

Three predominant factors make the resource-based constraint scheduling problem difficult are constraints, optimization, and search methods. For example, in experiment scheduling, one technique is required to limit the search for all possible experiment schedules and another technique is needed to resolve conflicts that occur in resource assignment for the experiment.

There are essentially two types of constraints in resource-based scheduling: absolute and relative (temporal) constraints. An absolute constraint is an activity or resource that is limited these limitations cannot be violated. For instance, due to the power considerations of the spacecraft, an instrument may only operate during spacecraft sunlight or an instrument may not observe a star when it is occulted by the Earth. A relative constraint describes the relation or relative position between two entities. An example of a relative constraint is when experiment A must be performed before experiment B or experiment A can occur only during experiment C. Collectively, absolute and relative constraints serve to delimit the space of suitable schedules that are generated. However, due

to the large number and different types of interacting constraints which can occur, this alone makes conflict resolution of constraints a dynamic and difficult problem. Thus, it is important to represent explicitly and understand constraints to effectively resolve conflicts during the generation of partial schedules.

Since the scientific or operational context will vary from application to application, scheduling goals will also be different. These goals drive the criteria and selection of methods used to schedule activities and resources. Multiple criteria may require several different techniques to generate an optimal schedule. There may need to be compromises in determining whether a schedule is optimal when multiple criteria are used. The difficulty lies in determining the evaluation criteria required to judge the best schedule. Also, as the operational or scientific context of the application changes with time, scheduling goals may change, invalidating current optimization criteria. Therefore it is important to implement a scheduling system that is flexible and can be easily modified under changing conditions.

Scheduling is searching. At each step in the scheduling process, some decision is made about which activities or resources to schedule and when to schedule them. Because of the large number of possible choices, the effort required to search the space of possible schedules is typically exponential. Effective search requires the early identification of good partial schedules which must be judged for their potential for being beneficial to the overall schedule. This is complicated by the fact that scheduling conflicts may not be detected until many steps into the search. It is desirable then to identify a minimal number of past decisions to resolve the conflict, backup, and fix instead of discarding the schedule.

Constraints, optimization, and search methods inherent to resource-based scheduling do make solutions difficult. Since scheduling problems vary from one application to another and are rarely isomorphic to each other, each application will require different solutions and techniques. A solution depends on the characteristics of a given problem which involve: the type of activities (number, complexity, similarity, fixedness, fragmentability, co-dependencies...), the type of resources, time requirements, and goals of the schedule...[Geoffroy et al., 1990]. Also, when a solution is found it needs to be flexible, because most of the time the scheduling environment is constantly evolving from one state to another.

THE UARS SOLSTICE SCHEDULING PROBLEM

The scheduling application that is currently being examined is for the SOLSTICE instrument, a solar-stellar observing instrument scheduled for launch aboard the UARS (see Figure 1) in November 1991 and the EOS in 1997. The SOLSTICE resource-based constraint scheduling problem consists of coordinating SOLSTICE experimental activity with UARS spacecraft activity such as spacecraft attitude maneuvers or Tracking Data and Relay Satellite contacts, selecting experiments for solar and stellar observation periods and determining instrument platform slew periods and sharing SSPP (Solar Stellar Pointing Platform) resources with two additional UARS instruments. The SOLSTICE instrument has two prime scientific goals: to measure solar ultra violet irradiance with a long term accuracy better than 1% using a set of selected stars as the calibration source, and

to provide daily, high accuracy, solar spectra information for use with the atmospheric observing instruments. The SOLSTICE scheduling system must provide a solution for both long- and short-term scientific constraints.

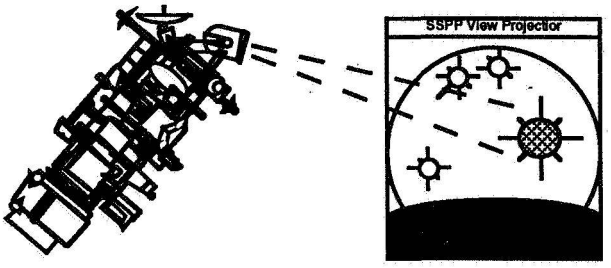

Figure 1 SSPP View from UARS Spacecraft

Scheduling the daily solar spectra activities and coordinating with the other two solar experiments on the SSPP is accomplished during the sunlight portion of the orbits. These activities include calibration of instrument pointing, collection of data for the daily solar spectra and meeting the requirements of the two other solar instruments while remaining within the constraints imposed by the UARS. Because the other two instruments can only view the sun, they place a high priority on the solar viewing resource of the SSPP. Managing the solar viewing resource for all of the instruments is also accomplished by the scheduler.

In scheduling stellar experiments, star observation statistics are used to select the experiments. Star observation statistics are the actual, scheduled and planned experiment data maintained on each star. A set of equations (see Figure 2) using star or target observation statistics is used to derive a star value or experiment priority. In addition, priorities for experiments are calculated based on the accessibility of the star, the success rate at which each star has been observed by the instrument, and an intrinsic star value assigned by the scientific investigator. After a priority is calculated, the experiment is scheduled by priority, star availability, and experiment duration. Stellar experiments are then selected to minimize slew time between stars, thereby optimizing the scientific return. This last step of optimization is to minimize slewing, which is very similar to the travelling salesman problem. It differs only in that the targets are moving with respect to each other over time.

STAR VALUE = TARGET VALUE + ACCESSIBILITY + SUCCESS RATE;

TARGET VALUE = V * (P/S);

where V = initial target value assigned by principal investigator;
S = total number of target observations;
P = number of attempted observations for a given target;

ACCESSIBILITY = K * (D/N);

where K = AC - (RAP - 1);
AC = total number of days for a given target availability period;
RAP = number of remaining days in current target availability period;
D = total number of days in the next unavailability period for a given target;
N = number of days to next unavailable period of given target;

SUCCESS RATE = C * V;

where C = number of actual target observations/total number of attempted target observations;
V = initial target value assigned by principal investigator;

Figure 2 Star Selection Equations

UARS spacecraft events influence the particular stellar or solar experiments scheduled. For instance, a spacecraft yaw around maneuver necessitates that only solar experiments be scheduled during this maneuver. The resulting schedule is an initial science plan with an optimal target acquisition sequence containing scheduled solar and stellar experiments.

AI TECHNIQUES CONSIDERED FOR THE UARS SOLSTICE APPLICATION

In the application of the scheduling problem for the SOLSTICE, several AI-based techniques were considered. There are several widely known AI-based techniques which have been applied in solving resource-

based constraint problems. These were examined and the following search-based techniques were found to be inappropriate for the UARS SOLSTICE scheduling application. These were neural nets, simulated annealing, and genetic algorithms. Neural networks have a reputation of working well in pattern recognition applications, but do not perform well in optimization problems. Simulated annealing and genetic algorithms are assured to produce good results provided appropriate assumptions are made. These two methods are theoretically interesting, but computationally intensive. These methods also select their solutions with some degree of randomness. These two approaches are suitable in certain areas of physics and biology, but not in resource-based scheduling problems for space applications.

Another AI-based search technique named "tabu search" applied well to the UARS SOLSTICE scheduling problem. This search has two significant advantages over the above methods. First, tabu search is easier than the other methods in implementation, and second, it has the advantage of being flexible in handling additional constraints as the scientific or operational requirements change.

TABU SEARCH METHODOLOGY

Tabu search is a proven method used for solving resource-based optimization problems. This methodology, as illustrated in Figure 3, can be applied to any operation which moves from one state to another. Each move is selected from a set of available moves which create a solution state. This solution state is evaluated to measure its relative value in a local sense. The search provides a guiding framework for generating the best global optimal solution after most classical searches reach a dead end at a local maximum. Tabu search prevents revisiting solutions which have previously been tried and probes for better solutions even after reaching a local maximum. Tabu search does this by using what is called a tabu list. The tabu list contains information regarding past moves or states which could lead to a solution already visited. This list is dynamically updated as the search progresses. If a move under consideration is found to be tabu, criteria are used to decide whether the move is good enough to override the tabu status. The criteria used in making this decision are called aspiration criterion. The aspiration criterion encourage the search for globally superior solutions instead of locally best solutions.

Tabu Search In The UARS SOLSTICE Application

The UARS SOLSTICE scheduling problem has been prototyped using the AI-based tabu search. A description of the algorithm follows. Tabu search starts with an initial schedule or solution state. For this application, the initial schedule is empty. This is also the least desirable schedule. Next, a candidate list of moves is generated. There are three types of moves: inserting, retracting and shifting of an experiment. Only one of these basic moves is applied to the current schedule to obtain a new schedule. For instance, with an empty schedule the candidate list of moves consists of inserting all possible experiments. However, after evaluating each insertion, a new schedule will contain only one experiment or move. The best admissible move among the list of candidate moves is found by evaluating or scoring each move in the candidate list.

An evaluation function scores each move in the candidate list and returns the score. This function is responsible for

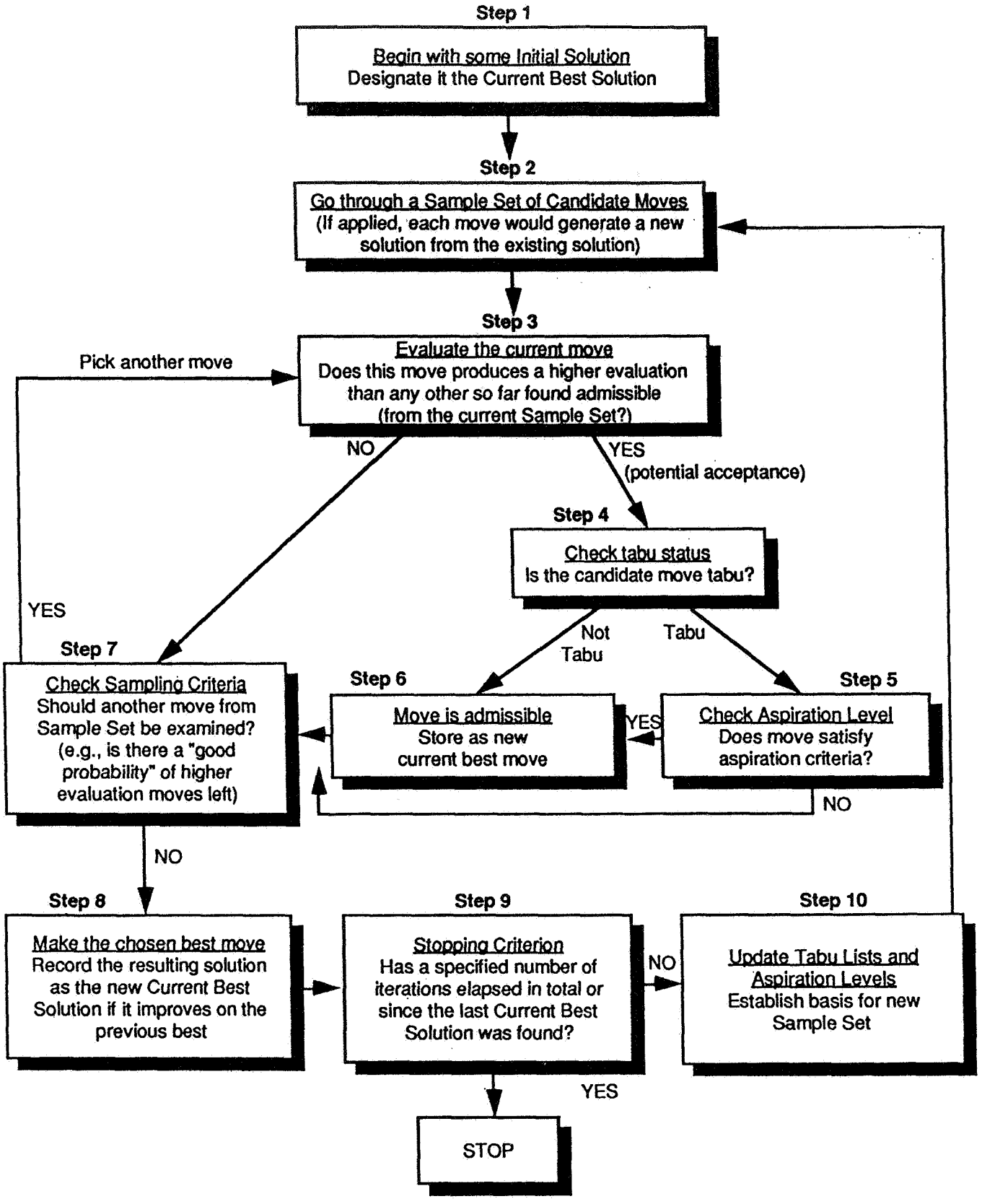

Figure 3 Tabu Search (used with author's permission: Glover, F., 1990)

ensuring that scientific goals and operational constraints are satisfied. Currently, several factors are considered in computing the score. First, the average priority of experiments for each star is considered. It is better to select a higher average priority which is derived from one experiment, than opposed to a lower average priority which is calculated from three experiments. Second, the slew time for targets are compared to the total experiment or observation time for all targets. If total observation time is greater or equal to total slew time then the score is increased. A third factor is scheduling a specified experiment where the experiment duration exceeds the target availability time. A small penalty is assessed because valuable scientific data is lost if the instrument's target is suddenly no longer available and the instrument is viewing deep space. The penalty is small because the experiment may be valuable enough to be considered for scheduling at some other time within the target availability period, thus replacing another experiment of a lesser value. These three factors maximize scientific return by ensuring that priority science is performed and that solar and stellar observing time is maximal. A fourth factor is slew backtracking. This is when a slew from one target to another is performed in the direction opposite to the spacecraft's flight direction. This is an operational constraint of the instrument and a large penalty is assessed to the score when this occurs.

After a move is evaluated, if the score is higher than any found admissible moves, then the next step is to check the tabu list. Checking the tabu list prevents moving to states already visited. If the move does not appear in the tabu list then it becomes the new current best move. However, if the move appears in the tabu list, the move is forbidden, or tabu, and cannot be inserted unless the aspiration criteria is satisfied. To satisfy the aspiration criteria the new value of the move must beat the one in the tabu list. If this is the case the move is permitted and becomes the new current best move. Each move inserted is evaluated until all moves in the list of candidate moves have been evaluated. If none of the moves evaluated is admissible or the schedule cannot be improved, the best schedule found so far is returned and the search is stopped.

If a specified number of iterations has elapsed since the best schedule has been found, then the search is stopped. Otherwise, in order to generate a new list of candidate moves, the tabu list is updated as well as the aspiration level. To update the tabu list, the move opposite the one added becomes tabu, thus resulting in the previous schedule. The aspiration criteria is updated with the new score and becomes the new score to beat. If the move added was previously tabu, then it is deleted from the tabu list. Once all updates are complete the next list of candidate moves is generated.

A Knowledge-based Approach Versus Tabu Search

The UARS SOLSTICE scheduling problem has been prototyped using two AI-based techniques: a knowledge-based approach and tabu search. In this section they are compared and discussed based upon the three factors (constraints, optimization, and search) discussed earlier.

The UARS SOLSTICE scheduling problem has been implemented using a knowledge-based approach. This prototype is operational and is currently being used in the Science User's Resource Expert (SURE), the AI-based scheduling component of the Science Users Resource Planning and Scheduling System (SURPASS). SURPASS is a software tool enabling

C-2

distributed planning and scheduling and is based on resource allocation and optimization [Thalman and Sparn, 1990]. In this knowledge-based approach, an expert system tool, CLIPS/Ada (C Language Integrated Production Shell/Ada) was used to implement an expert scheduler.

In the knowledge-based approach, constraints were easily represented by rules. Rules facilitate a natural representation of activities, resources, and constraints. CLIPS uses a LISP-like if-then language for rule construction. For example, an instrument operating constraint is to not make stellar observations which are within five degrees of the moon. This constraint easily translates into the English statement, "If a stellar observation is within five degrees of the moon, then do not observe". Rules can be inserted or removed as scientific objectives or mission constraints change. Scheduled activities such as solar or stellar observations were easily represented by templates or the more commonly known structure frames. In the tabu search methodology, constraints are represented in the evaluation function as variables. These variables are assigned a weight which influence the overall score. The difficulty is to derive an equation consisting of the various constraint variables to model a satisfactory solution. This takes time since a great deal of fine-tuning of the evaluation function is required in order to arrive at a practical solution.

In the knowledge-based approach, search and optimization of instrument activity were implemented by using and capturing planning and scheduling heuristics in rules. Since the number of possible experiments or schedules are too large to be able to consider all of them, schedules were generated using rules. These rules reflect the requests of scientists, instrument engineers and operators, and instrument planners and schedulers. Searching which is iterative is not naturally implemented using rules. For example, in selecting a stellar experiment where there were over 300 experiments possible, experiments were selected based on priority and duration. The CLIPS/Ada language provides the capability to implement looping using rules, but performance was somewhat slow. However, since the choice of implementation remains uncertain due to the continuing evolution of strategy and constraints, the rule-based looping might still be legitimate. Search and optimization are inherent to tabu search. These two criteria are the strengths of the algorithm. The tabu search prototype is implemented in Ada. Searching is based on creating a series of moves which consist of randomly selecting experiments and then evaluating the worth of the moves or current schedule. Since the generation of schedules was slow at first, heuristics were added to enhance performance. Each schedule is evaluated in order to arrive at the best global solution. No additional optimization need be built into the algorithm, except for tuning the evaluation function.

Results

Results of the tabu prototype have been successful in that it has allowed a detailed understanding of tabu search, the implementation was straight forward, and the results were good solutions. In terms of time, though, the performance of the prototype was slow. In comparing the results of tabu search to the knowledge-based methodology, it took approximately 30 minutes to schedule one hour of stellar observations using tabu search as opposed to five minutes using rules. At the time of this writing, improvements continue to be implemented to increase efficiency as well as improving the results. One such improvement is to start with an initial

schedule that contains one experiment per availability period, instead of starting with an empty schedule and inserting an experiment one by one. Since it involves time to insert, performance will most likely improve. Another improvement includes refining the heuristics throughout the code. Refined heuristics are needed to restrict the set of moves or experiments considered in order to improve efficiency. Also, the evaluation function requires more thought such as adjusting the weights of the factors used in the function. A fine-tuned function gives improved scores, thereby reducing the number of moves to be evaluated and increasing performance time.

CONCLUSION

Similarities were noticed during the development the UARS SOLSTICE scheduling application using different AI methodologies. Our previous experience of knowledge-based systems and our recent experience with tabu search showed us that both methods allowed scheduling heuristics and constraints to be easily added or removed as the problem definition evolved. This is important since most applications are in a state of flux from the initial conception to the operational stage and beyond. Scheduling systems must be flexible to changing conditions and be kept current so that functionality is not lost and the system does not become obsolete.

Traditionally, space scheduling applications have focused on scheduling in the operational context without considering, if applicable, the scientific goals. This concept is dated. Today, AI technology not only allows operational environments to be represented, but scientific objectives are easily represented and incorporated as well. Since the overall idea in scientific-oriented missions is to maximize scientific return, this should be the primary goal. Spacecraft and instrument performance should be maximized to satisfy and support scientific goals.

It is important to prototype realistic scheduling problems. Prototypes implemented using AI techniques provide valuable information in terms of ease of implementation, performance, and correctness of results. However, it is only after these prototypes become operational that we can learn if techniques used are productive and effective throughout the lifetime of the application.

REFERENCES

1. Geoffroy, A.L., Britt, D.L., Gohring, J.R., The Role of Artificial Intelligence In Scheduling Systems. NASA Conference Publication 3068, 1990 Goddard Conference on Space Applications of Artificial Intelligence, May 1-2, 1990.

2. Glover, F., Tabu Search: A Tutorial. Interfaces 20: July - August, 1990.

3. Ibaraki, T., Katoh, N., Resource Allocation Problems: Algorithmic Approaches. Cambridge, MA, MIT Press.

4. Kaufmann, A., Desbazeille, The Critical Path Method. Gordon and Breach Science Publishers, 1969.

5. Thalman, N.E., Sparn, T.P., Science User Resource Expert (SURE): A Science Planning and Scheduling Assistant For a Resource Based Environment. NASA Conference Publication 3068, 1990 Goddard Conference on Space Applications of Artificial Intelligence, May 1-2, 1990.

6. Wiest, J.D., Levy, F.K., A Management Guide to PERT/CPM. Prentice/Hall, 1969.